Laboratory Report
On
Open-ended Experiment-1 (Experiment-09)

Analyzing Network Performance in a Simple Wired Topology

Computer Networks Laboratory (CS39003)

Submitted by

2230142	Ajit Tripathy
2230148	Anmol Subham
2230162	Avilasha Goswami
2230167	Deep Habiswashi

Date of Submission	04-04-2025

Group: 1, ECSc-1 (2022 Admitted Batch)

B.Tech Programme in Electronics and Computer Science Engineering

School of Electronics Engineering
Kalinga Institute of Industrial Technology, Deemed to be University
Bhubaneswar, India
April 2025

AIM OF THE EXPERIMENT

The aim of this experiment is to simulate network performance in NS2 (Network Simulator 2) by analyzing key metrics such as throughput, packet delivery ratio, and network behavior under different conditions. The experiment includes evaluating TCP and UDP traffic, experimenting with various network topologies, and implementing mobility models to assess dynamic network performance.

REQUIREMENTS

- **Software:**
    - NS2 (Network Simulator Version 2)
    - Linux-based environment (Ubuntu 18.04 recommended, Docker setup available)
    - AWK scripting for data extraction and analysis
    - Gnuplot for visualization
- **Hardware:**
    - A system capable of running network simulations efficiently (4GB+ RAM recommended)

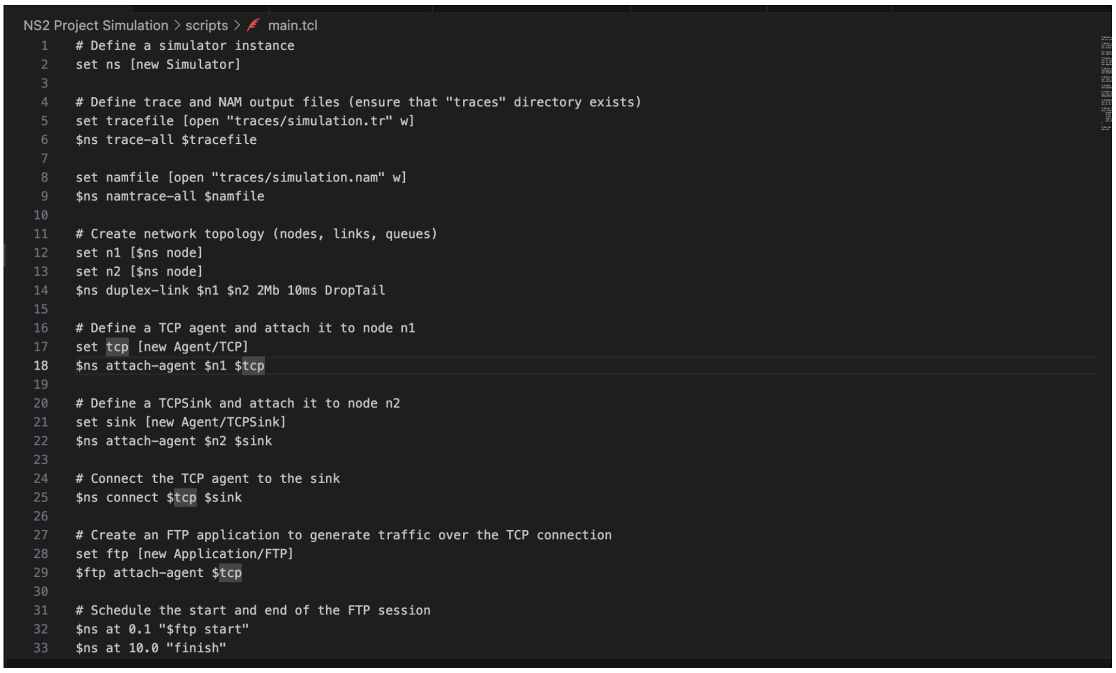
THEORY

NS2 (Network Simulator 2) is a discrete event-driven simulator designed to study network protocols, routing, and mobility models. It supports various network types, including wired and wireless networks. Key concepts covered in this project include:

- Packet Transmission and Reception: Simulation of how packets are sent and received across a network.
- Throughput Calculation: Measurement of data successfully transferred per unit time (kbps).
- Packet Delivery Ratio (PDR): The ratio of packets successfully received to packets sent.
- TCP vs UDP Traffic: Comparing the performance of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) under different network conditions.
- Mobility Models: Implementing models like Random Waypoint to simulate mobile network scenarios.

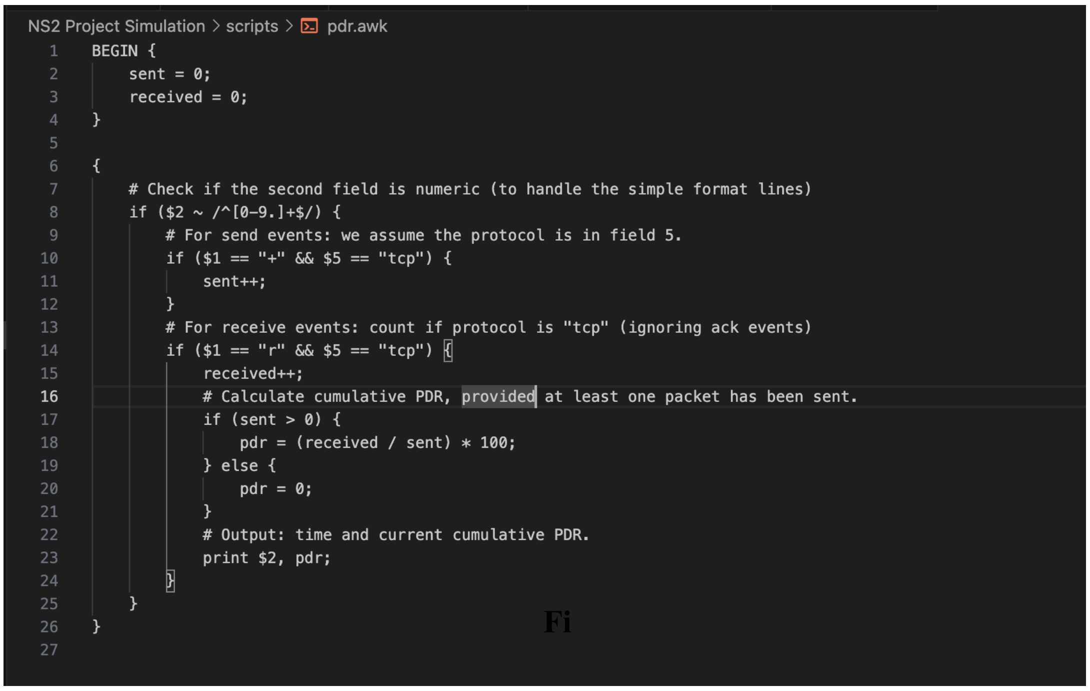
<u>CODE DEVELOPMENT</u>

1. TCL Script Development
   - **main.tcl**: Defines the network topology, nodes, links, and traffic patterns.
   - **test.tcl:** Implements test scenarios with varying parameters.
2. Trace File Analysis (AWK Scripts)
   - **throughput.awk:** Extracts throughput data from the .tr trace file.
   - **pdr.awk:** Computes packet delivery ratio from the trace file.
3. Visualization (Gnuplot)
   - **pdr_plot.plt:** Generates graphs for PDR analysis.
   - Additional script for throughput visualization.
4. Docker Environment Setup (if using Docker)
   - Commands for setting up the container and running



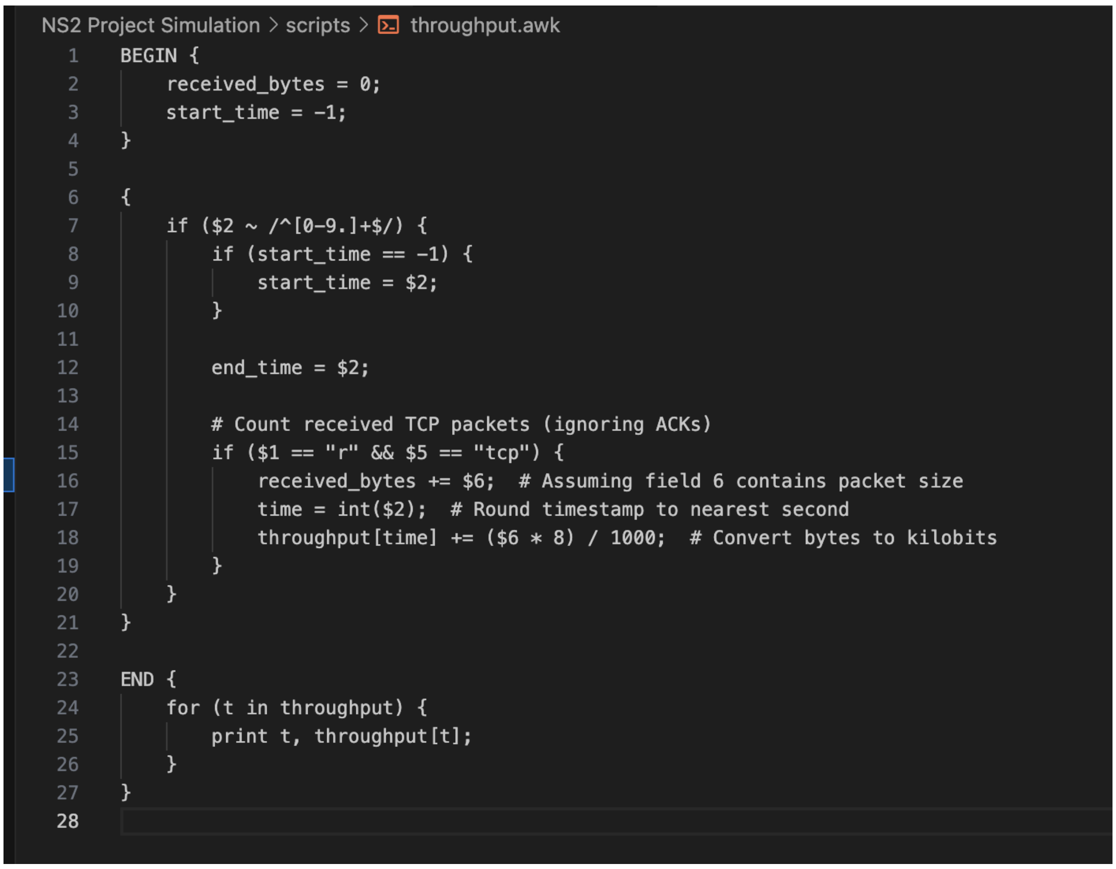
```
# Define a simulator instance
set ns [new Simulator]

# Define trace and NAM output files (ensure that "traces" directory exists)
set tracefile [open "traces/simulation.tr" w]
$ns trace-all $tracefile

set namfile [open "traces/simulation.nam" w]
$ns namtrace-all $namfile

# Create network topology (nodes, links, queues)
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n1 $n2 2Mb 10ms DropTail

# Define a TCP agent and attach it to node n1
set tcp [new Agent/TCP]
$ns attach-agent $n1 $tcp

# Define a TCPSink and attach it to node n2
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink

# Connect the TCP agent to the sink
$ns connect $tcp $sink

# Create an FTP application to generate traffic over the TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp

# Schedule the start and end of the FTP session
$ns at 0.1 "$ftp start"
$ns at 10.0 "finish"
```

. Fig. main.tcl script


NS2 Project Simulation > scripts > pdr.awk

```
BEGIN {
    sent = 0;
    received = 0;
}

{
    # Check if the second field is numeric (to handle the simple format lines)
    if ($2 ~ /^[0-9.]+$/) {
        # For send events: we assume the protocol is in field 5.
        if ($1 == "+" && $5 == "tcp") {
            sent++;
        }
        # For receive events: count if protocol is "tcp" (ignoring ack events)
        if ($1 == "r" && $5 == "tcp") {
            received++;
            # Calculate cumulative PDR, provided at least one packet has been sent.
            if (sent > 0) {
                pdr = (received / sent) * 100;
            } else {
                pdr = 0;
            }
            # Output: time and current cumulative PDR.
            print $2, pdr;
        }
    }
}
```

Fig. pdr.awk script

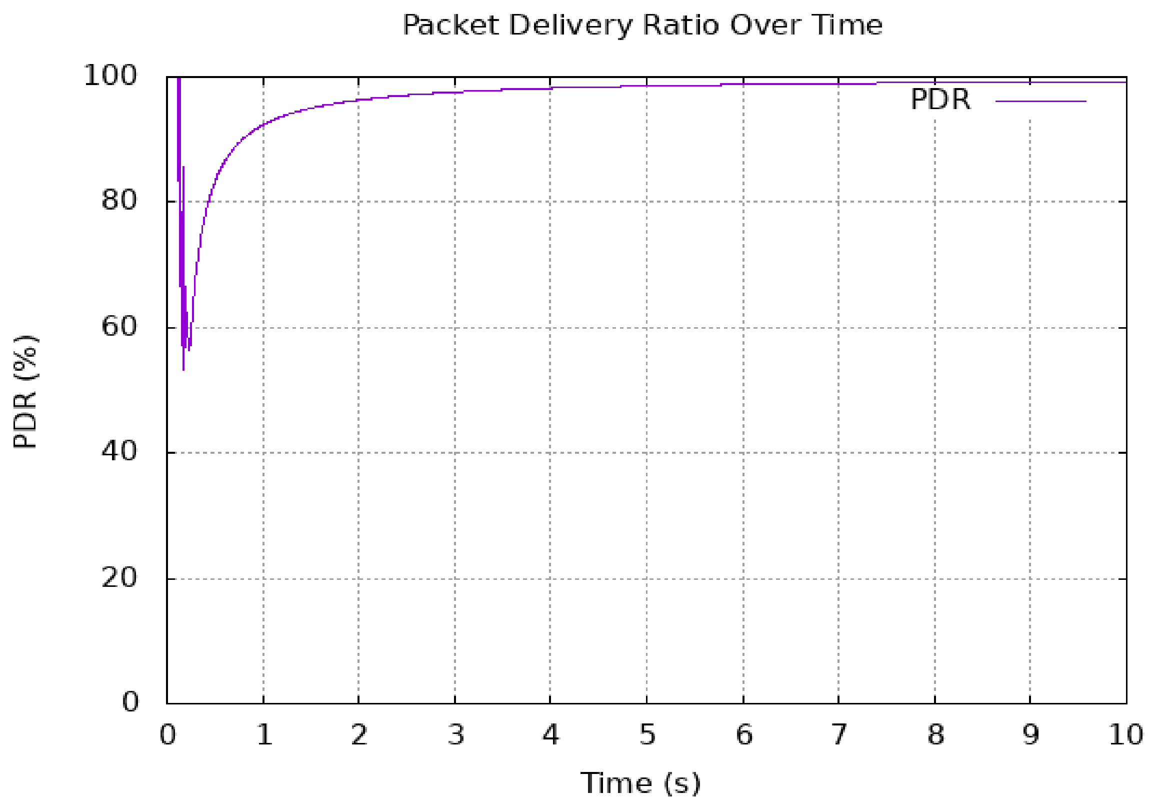

NS2 Project Simulation > scripts > throughput.awk

```
BEGIN {
    received_bytes = 0;
    start_time = -1;
}

{
    if ($2 ~ /^[0-9.]+$/) {
        if (start_time == -1) {
            start_time = $2;
        }

        end_time = $2;

        # Count received TCP packets (ignoring ACKs)
        if ($1 == "r" && $5 == "tcp") {
            received_bytes += $6;  # Assuming field 6 contains packet size
            time = int($2);  # Round timestamp to nearest second
            throughput[time] += ($6 * 8) / 1000;  # Convert bytes to kilobits
        }
    }
}

END {
    for (t in throughput) {
        print t, throughput[t];
    }
}
```

Fig. throughput.awk script

<u>OBSERVATIONS/RESULTS/GRAPHS</u>

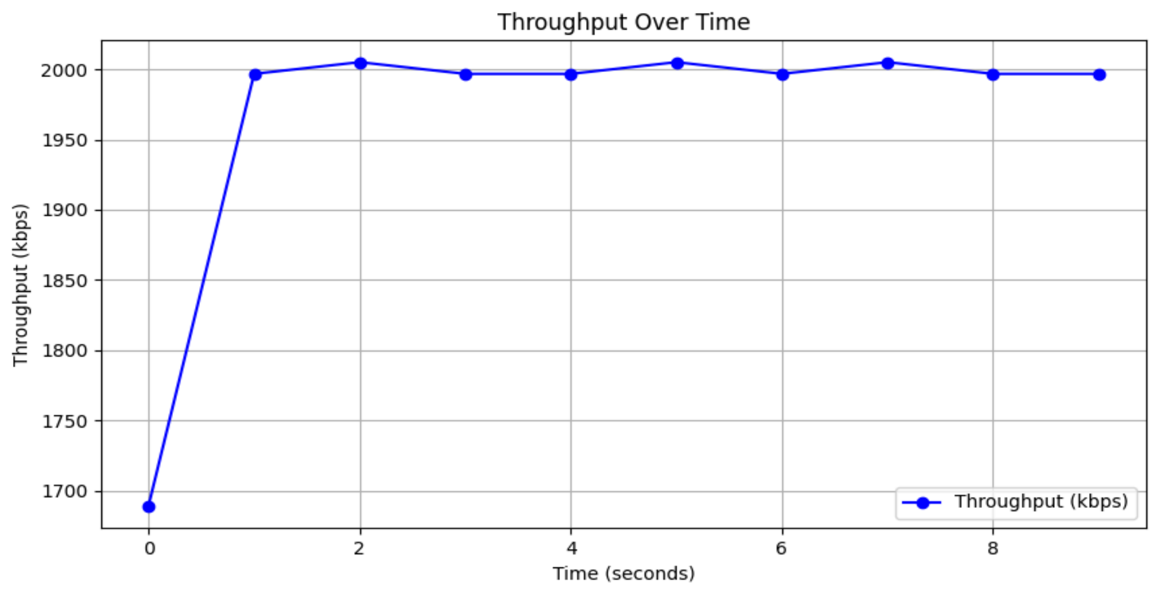
- Throughput Analysis
  - Measured throughput in kbps.
  - Comparison of different scenarios (e.g., TCP vs UDP, different topologies).
- Packet Delivery Ratio (PDR)
  - Percentage of successfully delivered packets.
  - Graphs generated using Gnuplot.
- Effect of Mobility Models
  - Performance analysis under static and dynamic network conditions.



Packet Delivery Ratio (PDR) Over Time

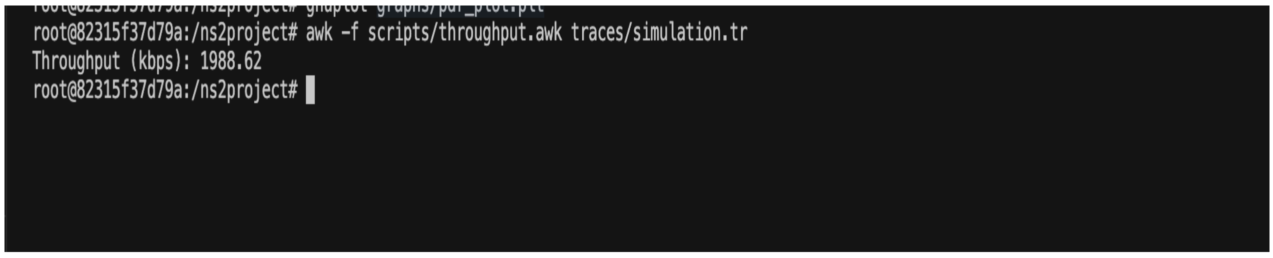
- This graph shows Packet Delivery Ratio (PDR) as a percentage over time.

- The curve starts low and rapidly increases, approaching 100%.
- This suggests that initially, some packets are lost due to network initialization, but the system stabilizes, achieving near-perfect packet delivery.



Throughput Over Time

- The graph represents network throughput (in kbps) over time (seconds).
- The sharp increase at the start indicates the network stabilizing as packets start flowing.
- After the initial rise, throughput remains nearly constant, fluctuating slightly around 2000 kbps, indicating a stable network performance.



```
root@82315f37d79a:/ns2project# gnuplot graphs/pdr_plot.plt
root@82315f37d79a:/ns2project# awk -f scripts/throughput.awk traces/simulation.tr
Throughput (kbps): 1988.62
root@82315f37d79a:/ns2project# 
```

Throughput Calculation Output (Terminal Screenshot)

- The terminal command executed (awk -f scripts/throughput.awk traces/simulation.tr) processes the NS2 trace file to calculate throughput.

- The printed output (Throughput (kbps): 1988.62) confirms that the throughput computed matches the values seen in the graph.

CONCLUSION:

This project successfully simulates network performance using NS2. The experiment highlights how different traffic types (TCP vs UDP) behave in a simulated environment. It also demonstrates the impact of mobility on network performance. The results show that NS2 is a powerful tool for network research, allowing detailed analysis of throughput, packet delivery, and overall efficiency. Future improvements can include more advanced mobility models and real-world network emulation techniques.

STUDENT SIGNATURE

Roll Number	**Name**	**Signature**
2230142		
2230148		
2230162		
2230167		

Date:-

SIGNATURE OF THE LAB FACULTY MEMBER

Faculty Name	Signature	Date